Adaptive Pseudo-Free Groups and Applications

Dario Fiore

École Normale Supérieure, Paris, France

DoE CRYPTODOC - Darmstadt, *November 21, 2011*

Joint work with: Dario Catalano (University of Catania, Italy)
Bogdan Warinschi (University of Bristol, UK)

This talk

1. Motivations
2. Intro: free and pseudo-free groups
3. Our contribution
   - Adaptive Pseudo-Free groups
   - Applications: signatures
   - Existence: $\mathbb{Z}_N^*$ is APF
4. Conclusions

This talk

1. Motivations
2. Intro: free and pseudo-free groups
3. Our contribution
   - Adaptive Pseudo-Free groups
   - Applications: signatures
   - Existence: $\mathbb{Z}_N^*$ is APF
4. Conclusions

From Theory to Practice

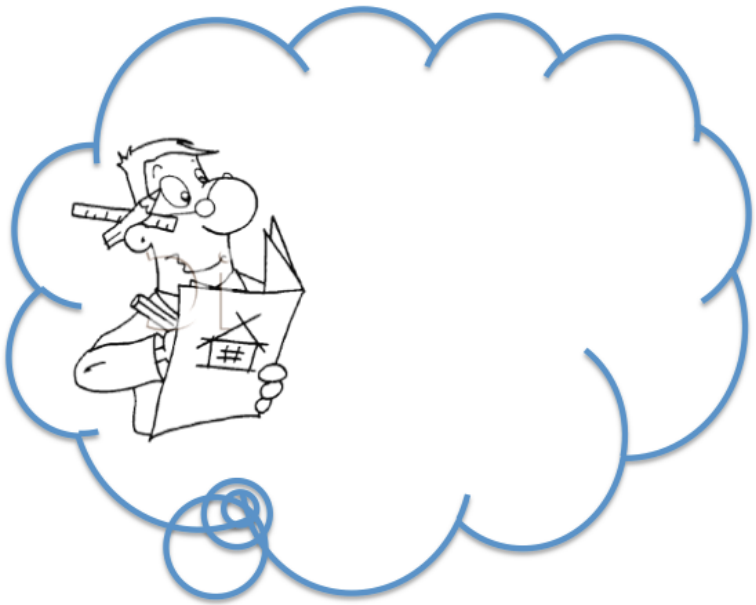
Abstractions in cryptography and security

- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusable, scalable proofs

Abstractions in cryptography and security

- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusab

1. Define an abstract model for some security guarantees



Abstractions in cryptography and security

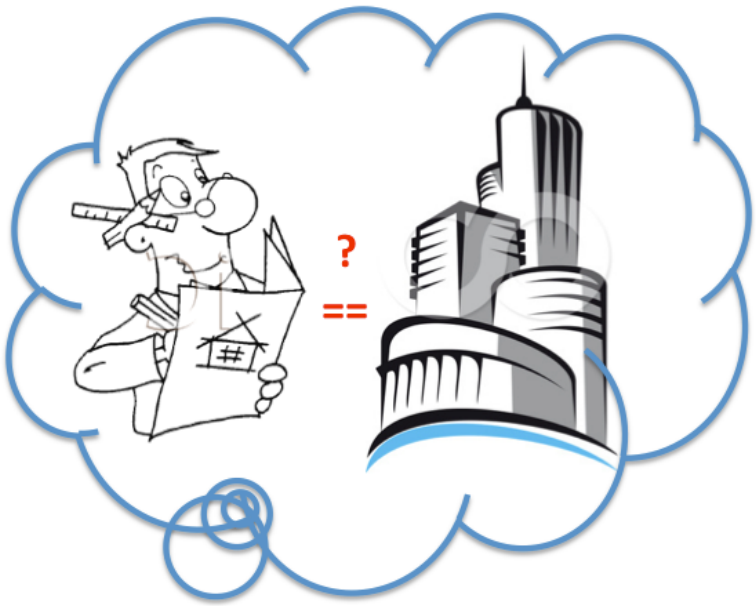
- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusab

1. Define an abstract model for some security guarantees
2. Instantiate a primitive

Abstractions in cryptography and security

- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusab

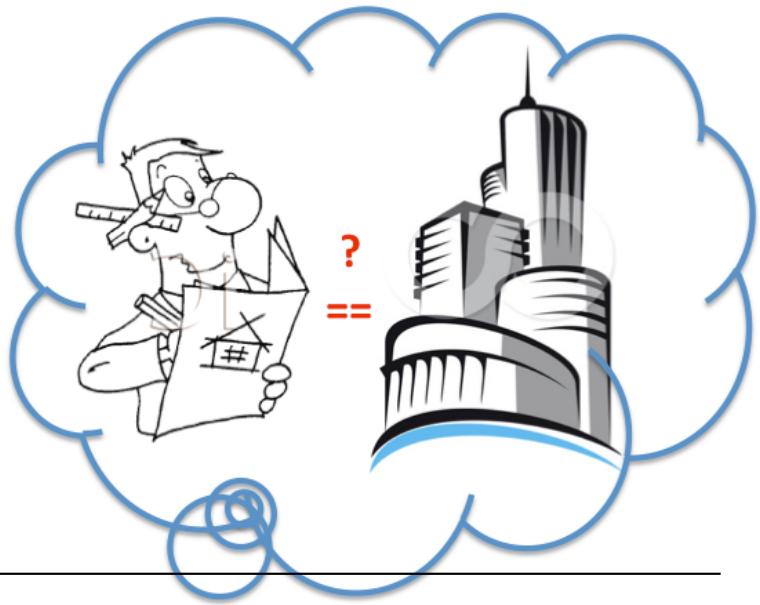
1. Define an abstract model for some security guarantees
2. Instantiate a primitive
3. Prove that the primitive satisfies the model



Abstractions in cryptography and security

- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusab

1. Define an abstract model for some security guarantees
2. Instantiate a primitive
3. Prove that the primitive satisfies the model



⇓

The instantiation is "secure"

Abstractions in cryptography and security

- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusable, scalable proofs

Examples

- Universal Composability
- Dolev-Yao

Abstractions in cryptography and security

- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusable, scalable proofs

Examples

- Universal Composability
- Dolev-Yao

- Most of them are not concerned about the mathematical structures underlying crypto constructions

Abstractions in cryptography and security

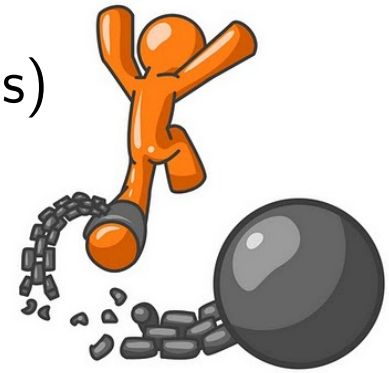
- Abstractions capture the essential security properties of primitives and protocols
- They allow for modular, reusable, scalable proofs

Examples

- Universal Composability
- Dolev-Yao

- Most of them are not concerned about the mathematical structures underlying crypto constructions
- One exception: Pseudo-Free Groups [Rivest04]

Free Groups



- Let $A = \{a, b, c, \ldots\}$ be a set of generators (symbols)
- Group operation: concatenation
- Identity: empty string ϵ
- Inverses: symbols $\{a^{-1}, b^{-1}, c^{-1}, \ldots\}$
- $\mathcal{F}(A) = \{a, aa, \ldots, ab, abb, \ldots, ab^{-1}c, \ldots\}$, all possible finite sequences of symbols
- We consider commutative, abelian free groups
- **A free group has infinite order**

They're "simple" objects!

Pseudo-Free Groups

- Introduced by Hohenberger in 2003 [Master's thesis] and later refined by Rivest [TCC04].

Intuition

A computational group is pseudo-free if it "behaves" as a free group to a computationally bounded machine.

Intuition

Consider the free group generated only by the symbol a,

$$\mathcal{F}(\{a\}) = \{\epsilon, a, aa, aaa, aaaa, \ldots\}$$

Consider the following equations

- $x^2 = a^4$
  It has solution $x = a^2$. We say it is *trivial* in $\mathcal{F}(\{a\})$

Intuition

Consider the free group generated only by the symbol a,

$$\mathcal{F}(\{a\}) = \{\epsilon, a, aa, aaa, aaaa, \ldots\}$$

Consider the following equations

- $x^2 = a^4$
  It has solution $x = a^2$. We say it is *trivial* in $\mathcal{F}(\{a\})$
- $x^2 = a$
  It has NO solutions in $\mathcal{F}(\{a\})$.

Intuition

Consider the free group generated only by the symbol a,

$$\mathcal{F}(\{a\}) = \{\epsilon, a, aa, aaa, aaaa, \ldots\}$$

Consider the following equations

- $x^2 = a^4$
  It has solution $x = a^2$. We say it is *trivial* in $\mathcal{F}(\{a\})$
- $x^2 = a$
  It has NO solutions in $\mathcal{F}(\{a\})$.
- $x^2 = a \bmod N$ in $\mathbb{Z}_N^*$ (where $N = pq$)
  It has solution if a is a square, but it cannot be *efficiently computed* without knowing $ord(\mathbb{Z}_N^*) = \phi(N)$.

Intuition

Consider the free group generated only by the symbol a,

$$\mathcal{F}(\{a\}) = \{\epsilon, a, aa, aaa, aaaa, \ldots\}$$

Cons

Intuition for pseudo-freeness

Free group	Computational group
Impossible	Computationally infeasible

- $x^2 = a \bmod N$ in $\mathbb{Z}_N^*$ (where $N = pq$)
  It has solution if a is a square, but it cannot be *efficiently computed* without knowing $ord(\mathbb{Z}_N^*) = \phi(N)$.

A formal definition: equations over free groups

- $X = \{x_1, \ldots, x_n\}$ set of variables
- $A = \{a_1, \ldots, a_m\}$ set of constant symbols
- Consider equations of the form

$$x_1^{e_1} x_2^{e_2} \cdots x_n^{e_n} = a_1^{s_1} a_2^{s_2} \cdots a_m^{s_m}$$

- e_i's and s_i's are integers.
- A **solution** in $\mathcal{F}(A)$ is an assignment $\psi : X \to \mathcal{F}(A)$ such that:

$$\psi(x_1)^{e_1} \cdots \psi(x_n)^{e_n} = a_1^{s_1} \cdots a_m^{s_m}$$

e.g., for $x^2 = a^4$, the solution is $\psi(x) = a^2$

A formal definition: equations over free groups

- $X = \{x_1, \ldots, x_n\}$ set of variables
- $A = \{a_1, \ldots, a_m\}$ set of constant symbols
- Consider equations of the form

$$x_1^{e_1} x_2^{e_2} \cdots x_n^{e_n} = a_1^{s_1} a_2^{s_2} \cdots a_m^{s_m}$$

- e_i's and s_i's are integers.
- A **solution** in $\mathcal{F}(A)$ is an assignment $\psi : X \to \mathcal{F}(A)$ such that:

$$\psi(x_1)^{e_1} \cdots \psi(x_n)^{e_n} = a_1^{s_1} \cdots a_m^{s_m}$$

Trivial equations

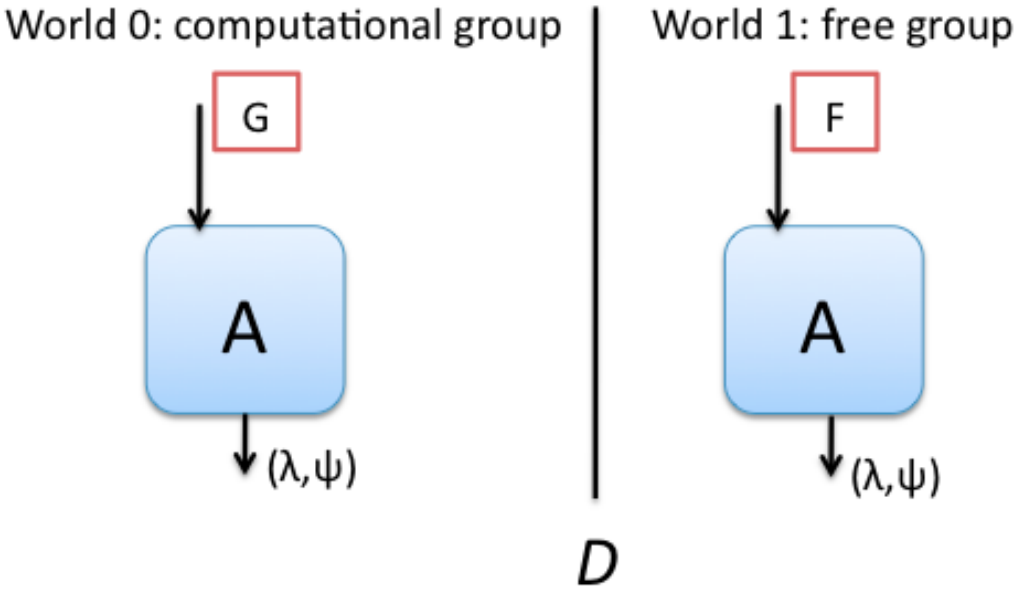
λ has solutions in $\mathcal{F}(A)$ (aka λ is *trivial*) if and only if

$$\forall i = 1, \ldots, m : gcd(e_1, \ldots, e_n) | s_i$$

Pseudo-free groups

Intuition

Free group	Computational group
Impossible	Computationally infeasible



(Static) Pseudo-Free Groups

Definition

A family of computational groups $\mathcal{G} = \{\mathbb{G}_n\}_n$ is *pseudo-free* if: for randomly chosen n, $|A| = poly(k)$ and any PPT $\mathcal{A}$:

$$\Pr[\mathcal{A}(\mathbb{G}_n, A) \rightarrow (\lambda, \psi)] \leq negl$$

such that:

1. λ has no solution in $\mathcal{F}(A)$ (i.e., λ is *non-trivial*)
2. λ has solution ψ in $\mathbb{G}_n$

Why are PF groups interesting?

Rivest showed that in a pseudo-free group a lot of assumptions naturally hold:

- Order problem: find e such that $a^e = 1$
- DLP: find x such that $a^x = b$
- RSA: find x such that $x^e = a$ (for $e > 1$)
- Strong RSA: find x, e such that $x^e = a$
- $\vdots$
- CDH: only in some restricted settings

Why are PF groups interesting?

PF groups might be a nice framework to do crypto!

Design crypto primitives in the free group
(*simple structure - impossible equations*)

⇓

Instantiate them in a computational PF group

⇓

We obtain a secure concrete instance

Interesting questions about pseudo-free groups

1. Do pseudo-free groups exist?
2. Is the notion of pseudo-freeness suitable for cryptographic applications?

Interesting questions about pseudo-free groups

1. Do pseudo-free groups exist?
2. Is the notion of pseudo-freeness suitable for cryptographic applications?

Question 1 ✔

- Rivest [TCC04] conjectured that the RSA group is pseudo-free
- Micciancio [Eurocrypt05] solved this problem showing that under the Strong-RSA assumption $\mathbb{Z}_N^*$ is pseudo-free when N is product of safe primes.

Interesting questions about pseudo-free groups (2)

1. Do pseudo-free groups exist?
2. Is the notion of pseudo-freeness suitable for cryptographic applications?

Question 2 ✗

- The current definition does not seem to be sufficient :-(
- Why? An adversary interacting with a cryptographic primitive (built upon a PF group) may obtain additional informations (e.g. solutions for other equations).
- This problem was recognized and left opened by Rivest

Example: digital signatures

The adversary can produce a forgery after having seen other signatures.

Our contribution

1. Notion of Adaptive Pseudo-Free (APF) Groups
2. Applications: APF groups can be used to build digital signatures and network coding (homomorphic) signatures.
3. APF groups exist: $\mathbb{Z}_N^*$ is APF under Strong-RSA

Corollary:

4. A framework for signatures based on Strong-RSA

Adaptive Pseudo-Freeness

Rough idea: The adversary can output a non-trivial equation with a solution after interacting with the group.

APF game

Setup Randomly choose a group $\mathbb{G}$ from the family, fix constants A and give $(\mathbb{G}, A)$ to $\mathcal{A}$.

Equation queries $\mathcal{A}$ is allowed to see non-trivial equations with their solutions.

Challenge $\mathcal{A}$ outputs (λ^*, ψ^*) and wins if λ^* is non-trivial and ψ^* is a solution in $\mathbb{G}$.

Adaptive Pseudo-Freeness

Rough idea: The adversary can output a non-trivial equation with a solution after interacting with the group.

APF game

Setup Randomly choose a group $\mathbb{G}$ from the family, fix constants A and give $(\mathbb{G}, A)$ to $\mathcal{A}$.

Equation queries $\mathcal{A}$ is allowed to see non-trivial equations with their solutions.

Challenge $\mathcal{A}$ outputs (λ^*, ψ^*) and wins if λ^* is non-trivial and ψ^* is a solution in $\mathbb{G}$.

Two challenging points:

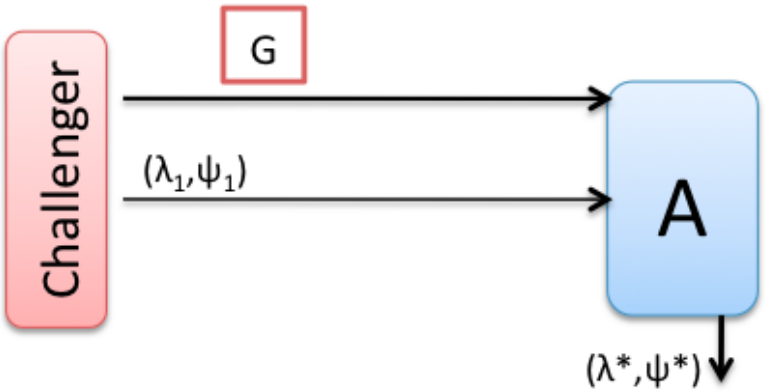
- is allowed to see non-trivial equations
- non-trivial w.r.t. other equations

How are the queried equations produced?

$\mathcal{A}$ is allowed to see non-trivial equations

Some possible solutions:

- The Challenger chooses equation λ_i and gives (λ_i, ψ_i) to $\mathcal{A}$.



How are the queried equations produced?

$\mathcal{A}$ is allowed to see non-trivial equations

Some possible solutions:

- The Challenger chooses equation λ_i and gives (λ_i, ψ_i) to $\mathcal{A}$.

Too weak and not really adaptive

How are the queried equations produced?

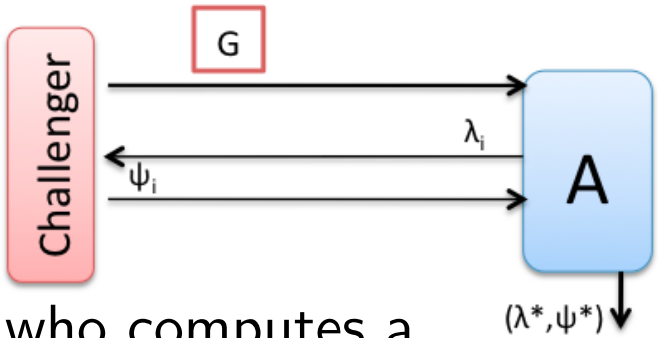
$\mathcal{A}$ is allowed to see non-trivial equations

Some possible solutions:

- The Challenger chooses equation λ_i and gives (λ_i, ψ_i) to $\mathcal{A}$.

Too weak and not really adaptive

- $\vdots$



- $\mathcal{A}$ chooses λ_i, gives it to the Challenger who computes a solution ψ_i for $\mathcal{A}$ (adaptively repeated)

How are the queried equations produced?

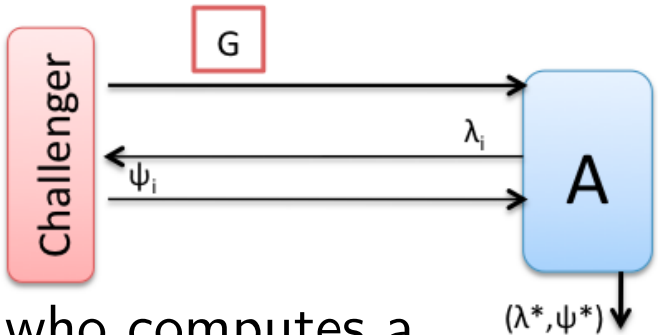
$\mathcal{A}$ is allowed to see non-trivial equations

Some possible solutions:

- The Challenger chooses equation λ_i and gives (λ_i, ψ_i) to $\mathcal{A}$.

Too weak and not really adaptive

- ⋮



- $\mathcal{A}$ chooses λ_i, gives it to the Challenger who computes a solution ψ_i for $\mathcal{A}$ (adaptively repeated)

Very strong! It seems unrealizable :-(

How are the queried equations produced?

$\mathcal{A}$ is allowed to see non-trivial equations

Some possible solutions:

- The Challenger chooses equation λ_i and gives (λ_i, ψ_i) to $\mathcal{A}$.

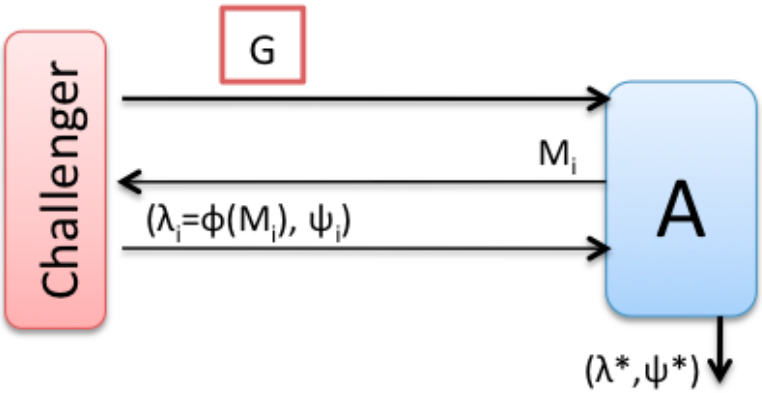
Too weak and not really adaptive

Solution: Parametric distribution

- $\mathcal{A}$ chooses λ_i, gives it to the Challenger who computes a solution ψ_i for $\mathcal{A}$ (adaptively repeated)

Very strong! It seems unrealizable :-(

A security notion parametrized by $\varphi(\cdot)$



A parametric distribution $\varphi(\cdot)$

λ_i is created according to $\varphi(M_i)$, conditioned on M_i chosen by $\mathcal{A}$.
$(e_1, \ldots, e_n, s_1, \ldots, s_m) \leftarrow \varphi(M)$

- φ "controls the power" of the adversary
- Different φ can capture the weak and strong definitions sketched before.
- Formally, we will say that $\mathcal{G}$ **is APF w.r.t. φ**
- We will define a sufficiently adaptive $\hat{\varphi}$ that allows for building signatures and proving APF of the RSA group.

Non-trivial equations in the adaptive setting

Non-trivial equation w.r.t. other equations

- Let $\Lambda = \{\lambda_k\}_{k=1}^{t}$ be the set of equations with solutions $\{\psi_k\}$ obtained by the adv.
- $\mathcal{A}$ might output $\lambda^* = \lambda_k$: it is non-trivial w.r.t. the old def.

Non-trivial equations in the adaptive setting

Non-trivial equation w.r.t. other equations

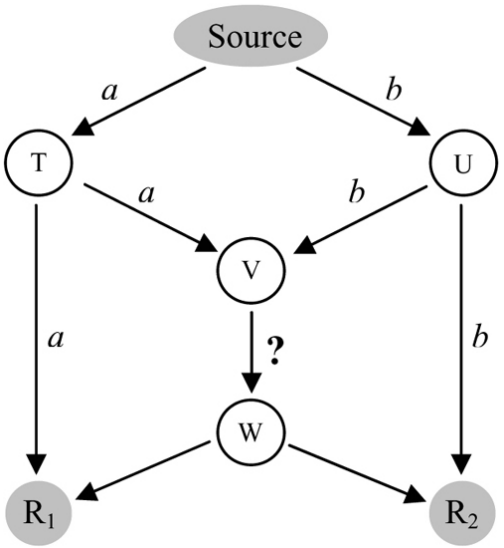
- Let $\Lambda = \{\lambda_k\}_{k=1}^{t}$ be the set of equations with solutions $\{\psi_k\}$ obtained by the adv.
- $\mathcal{A}$ might output $\lambda^* = \lambda_k$: it is non-trivial w.r.t. the old def.
- We define non-triviality by looking at the augmented free group $\mathcal{F}(A) \bigcup \{\psi_1(x), \ldots, \psi_t(x)\}$
- Another way to look at it: λ^* is trivial w.r.t. Λ if it can be obtained from a linear combination of equations in Λ.

See the formal definition in the paper.

Applications of APF groups

- **Network Coding (homomorphic) signatures**

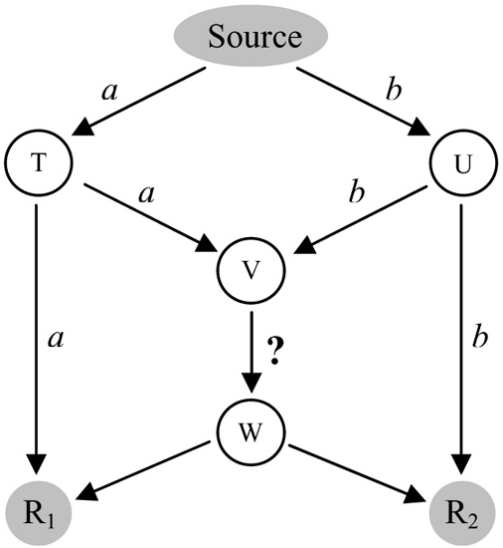
- **(Standard) Digital signatures**



Applications of APF groups

- **Network Coding (homomorphic) signatures**

- **(Standard) Digital signatures**



Network Coding homomorphic Signatures - Notion

They allow to sign n-dimensional vector subspaces.

- $\mathsf{NetKG}() = (\mathsf{vk}, \mathsf{sk})$: key generation
- $\mathsf{Sign}(\mathsf{sk}, \mathsf{fid}, W) = \sigma$: W= n-dimensional vector space; fid= unique identifier. W is described by a basis $(w^{(1)}, \cdots, w^{(n)})$. Usually, σ is a set of signatures σ_i on each $w^{(i)}$.
- $\mathsf{Ver}(\mathsf{vk}, \mathsf{fid}, w, \sigma) = 0/1$: 1 if σ valid for w.
- $\mathsf{Combine}(\mathsf{vk}, \mathsf{fid}, \{w_i, \sigma_i, \alpha_i\}_{i=1}^{\mu}) = \sigma$: σ correctly verifies $w = \sum_{i=1}^{\mu} \alpha_i \cdot w^{(i)}$.

Main application

Linear Network Coding.

Network Coding Signatures - Security Definition

- Similar to the usual unforgeability notion

Challenger	**Adversary**

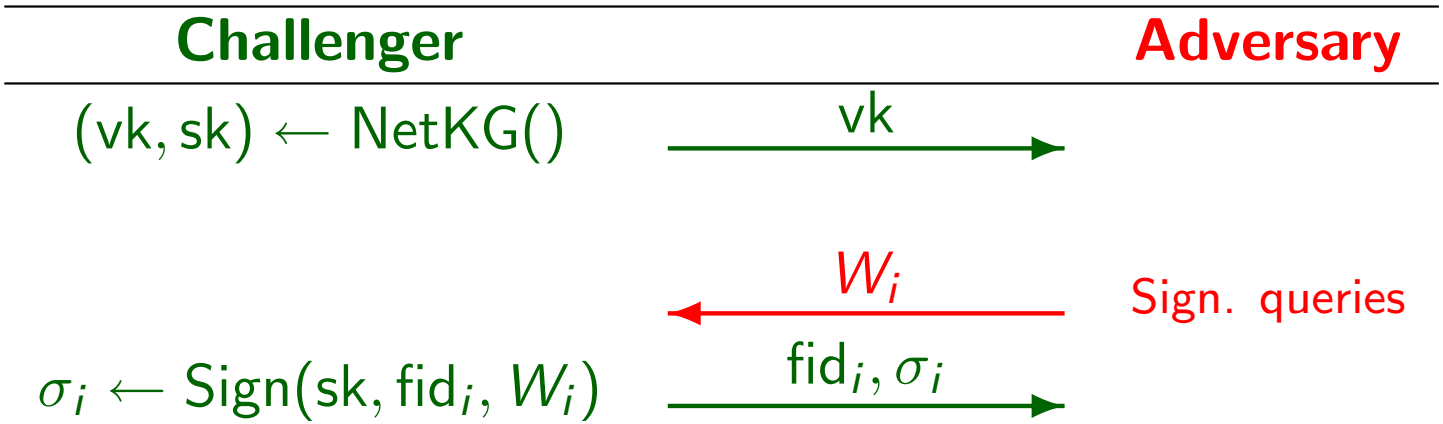
Network Coding Signatures - Security Definition

- Similar to the usual unforgeability notion

Challenger		**Adversary**
$(vk, sk) \leftarrow NetKG()$	$\xrightarrow{vk}$	

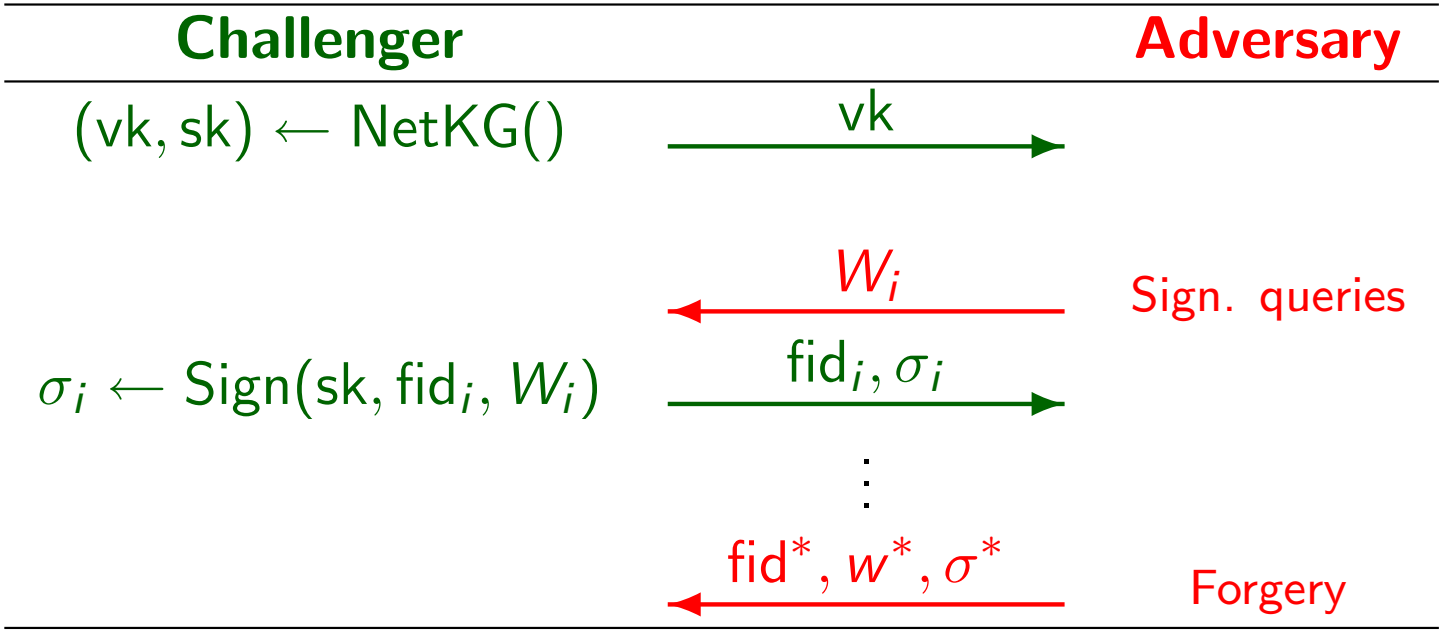
Network Coding Signatures - Security Definition

- Similar to the usual unforgeability notion



Network Coding Signatures - Security Definition

- Similar to the usual unforgeability notion



Adversary wins if σ^* valid for (w^*, fid^*), and:

1. $fid^* \neq fid_i$, $\forall i$
2. or, $fid^* = fid_j$, but $w^* \notin W_j$ (namely, σ^* is not a linear combination of signatures created by the Challenger).

Network Coding Signatures from APF groups

Preliminaries

- Assume equations in a canonical form $x^e = a_1^{s_1} \cdots a_m^{s_m}$

We design a class of parametric distributions $\varphi_{\ell,n}$

$\varphi_{\ell,n} : \mathbb{Z}_{2^\ell}^{n' \times n} \rightarrow \mathbb{Z}^{1+mn} \times \{0,1\}^*$ such that $(n + n' \leq m + 1)$

- Let fid be a binary string taken with high-entropy distribution;
- $e = H(\text{fid})$ where H is a map to primes
- For $i = 1$ to n define $\vec{s}^{(i)}$ as follows:
  - $s_i^{(i)} = 1$, $s_j^{(i)} = 0$, $\forall j = 1, \ldots, n : j \neq i$
  - $s_k^{(i)} \in \mathbb{Z}_e$ according to (arbitrary) D_k, $\forall k = n+1, \ldots, m$
- Output $(e, \text{fid}, \{\vec{s}^{(i)}\}_{i=1}^n)$. $\varphi_{\ell,n}$ defines n equations.

$\varphi_{\ell,n}$ is equipped with an efficient algorithm for verifying whether a tuple $(e, \text{fid}, \vec{s})$ is in the support of $w \in \mathbb{Z}_{2^\ell}^{n'}$ or not.

Network Coding Signatures from APF groups (2)

- Let $\mathcal{G}$ be a family of APF groups w.r.t. $\hat{\varphi} \in \varphi_{\ell,n}$:

NetKG(n): Randomly choose a group $\mathbb{G}$ from $\mathcal{G}$, fix sets X, A and set vk $= (X, A, \mathbb{G}, \varphi_{\ell,n})$ and sk $= ord(\mathbb{G})$

NetSign(sk, fid, W): $\{\lambda_i\}_{i=1}^{m} \leftarrow \hat{\varphi}(W)$. For $i = 1$ to m: use $ord(\mathbb{G})$ to find solution ψ_i for λ_i and output $\sigma_i = (\lambda_i, \psi_i)$

NetVer(vk, fid, w, σ) (1) Check that λ is distributed according to $\hat{\varphi}$. (2) Check that ψ is a valid solution.

Combine(vk, fid, $\{w_i, \sigma_i, \alpha_i\}_{i=1}^{\mu}$): output $\sigma = (\lambda, \psi)$ where $\psi = \prod_{i=1}^{\mu} \psi_i^{\alpha_i}$, $\quad \vec{s} = \sum_{i=1}^{\mu} \alpha_i \cdot \vec{s}^{(i)}$

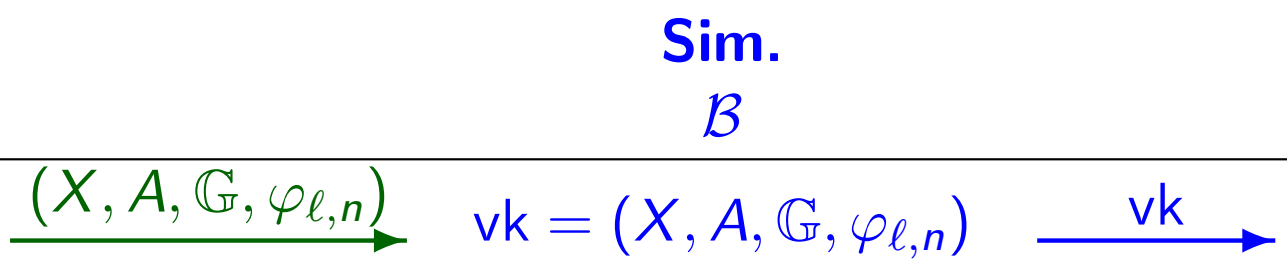
Theorem (Security of NetPFSig)

If $\mathcal{G}$ is APF w.r.t. $\hat{\varphi} \in \varphi_{\ell,n}$ ($\forall n \geq 1$), then NetPFSig(n) is secure.

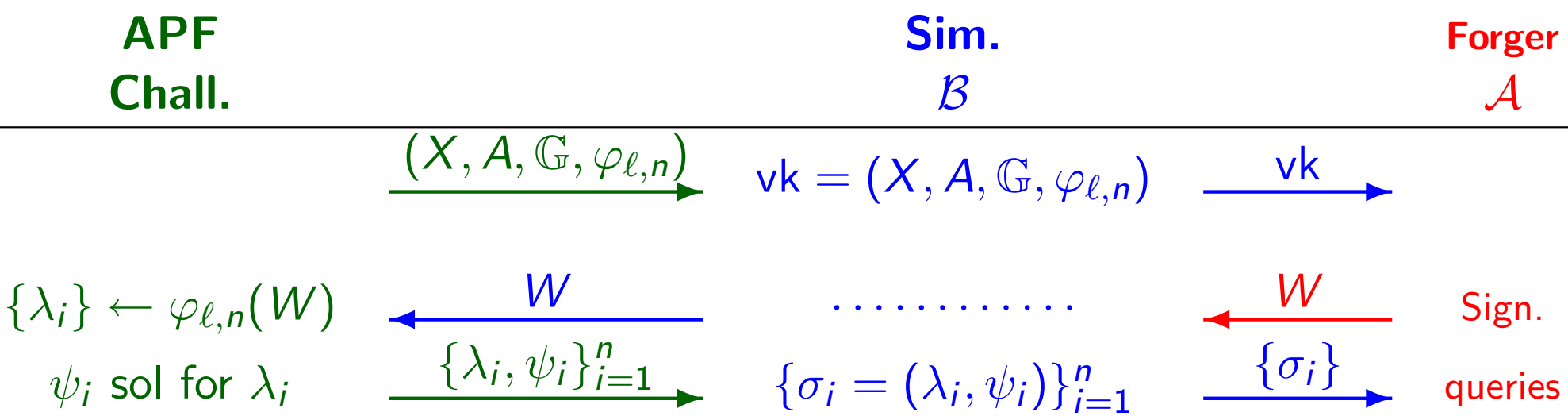
Proof (sketch)

APF	**Sim.**	**Forger**
Chall.	$\mathcal{B}$	$\mathcal{A}$

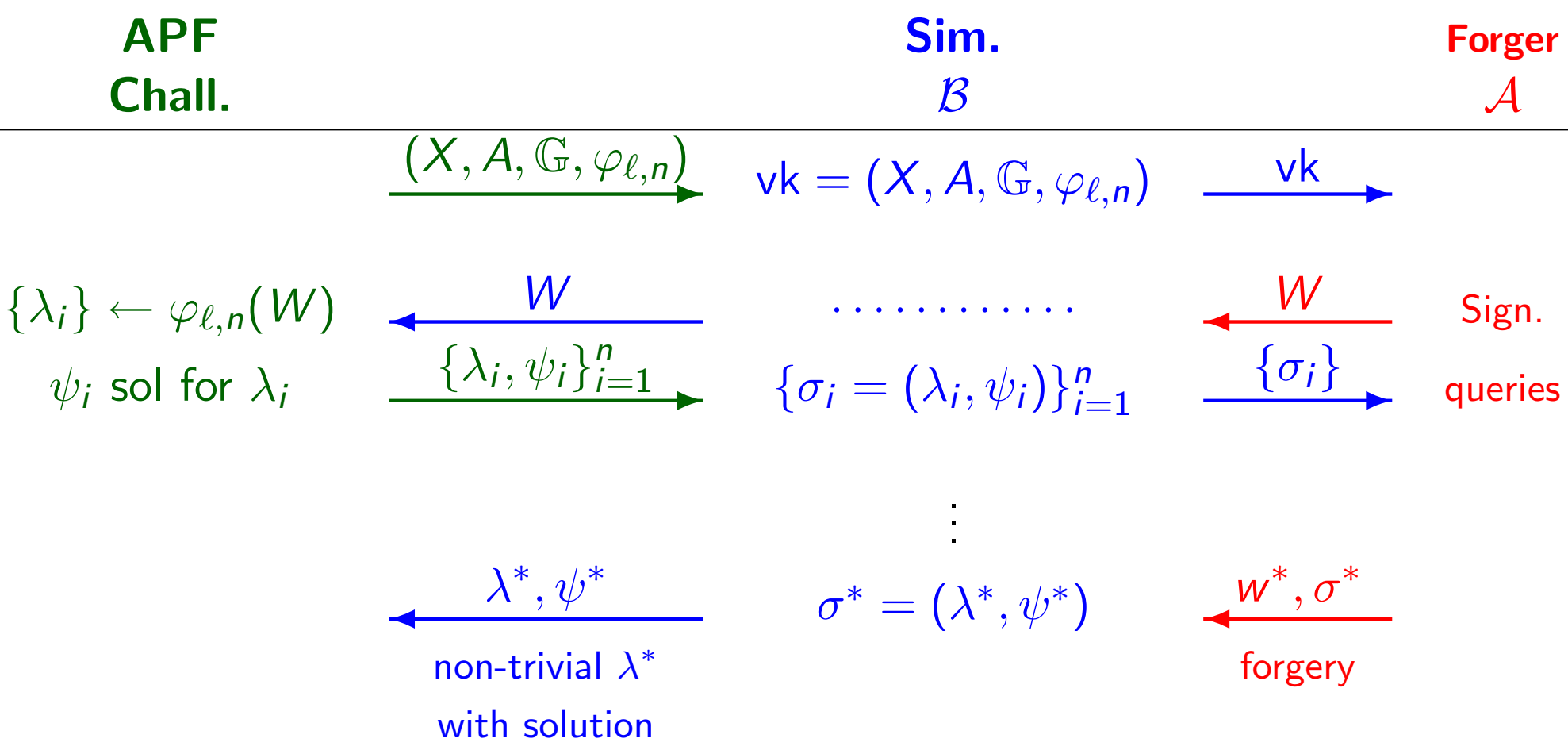
Proof (sketch)



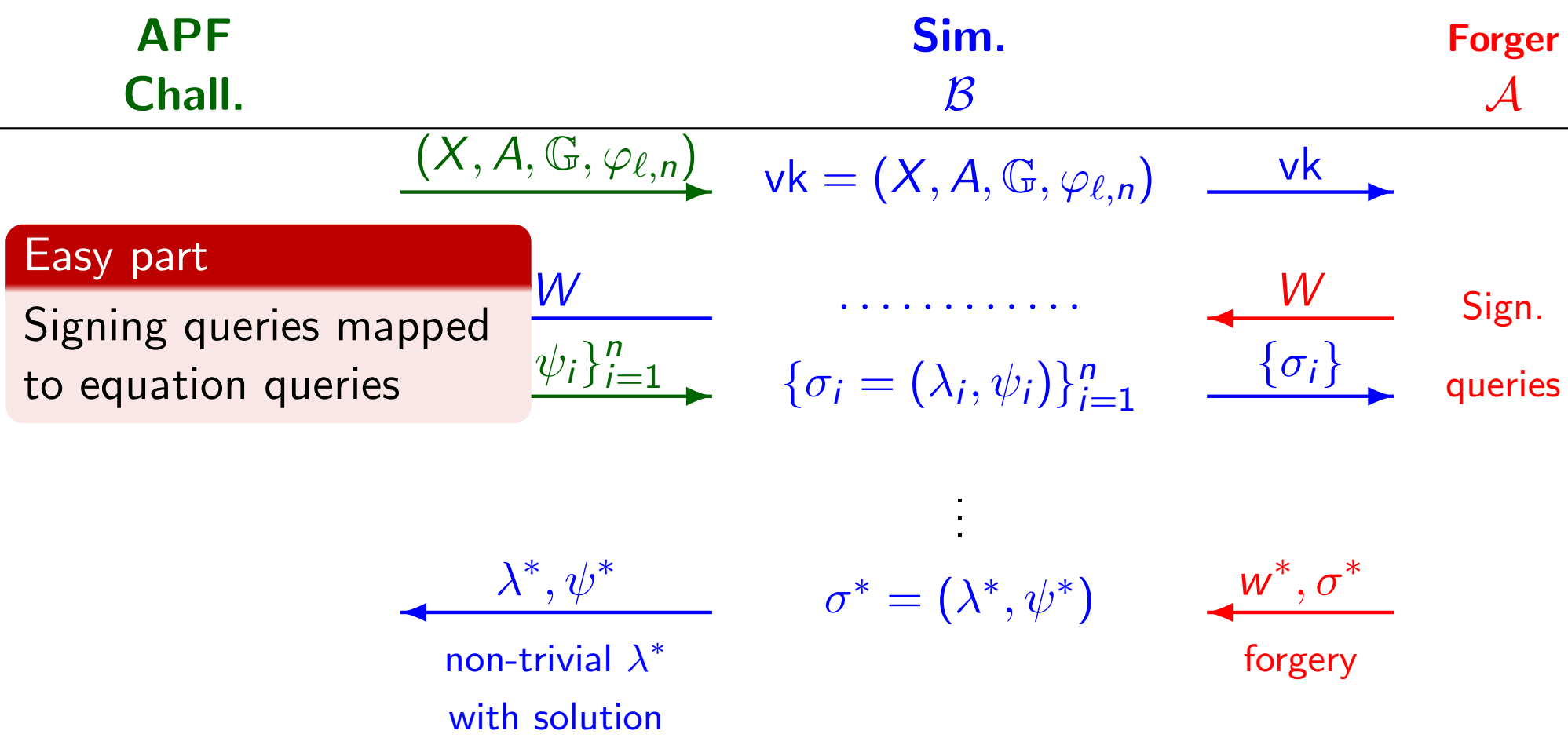
Proof (sketch)



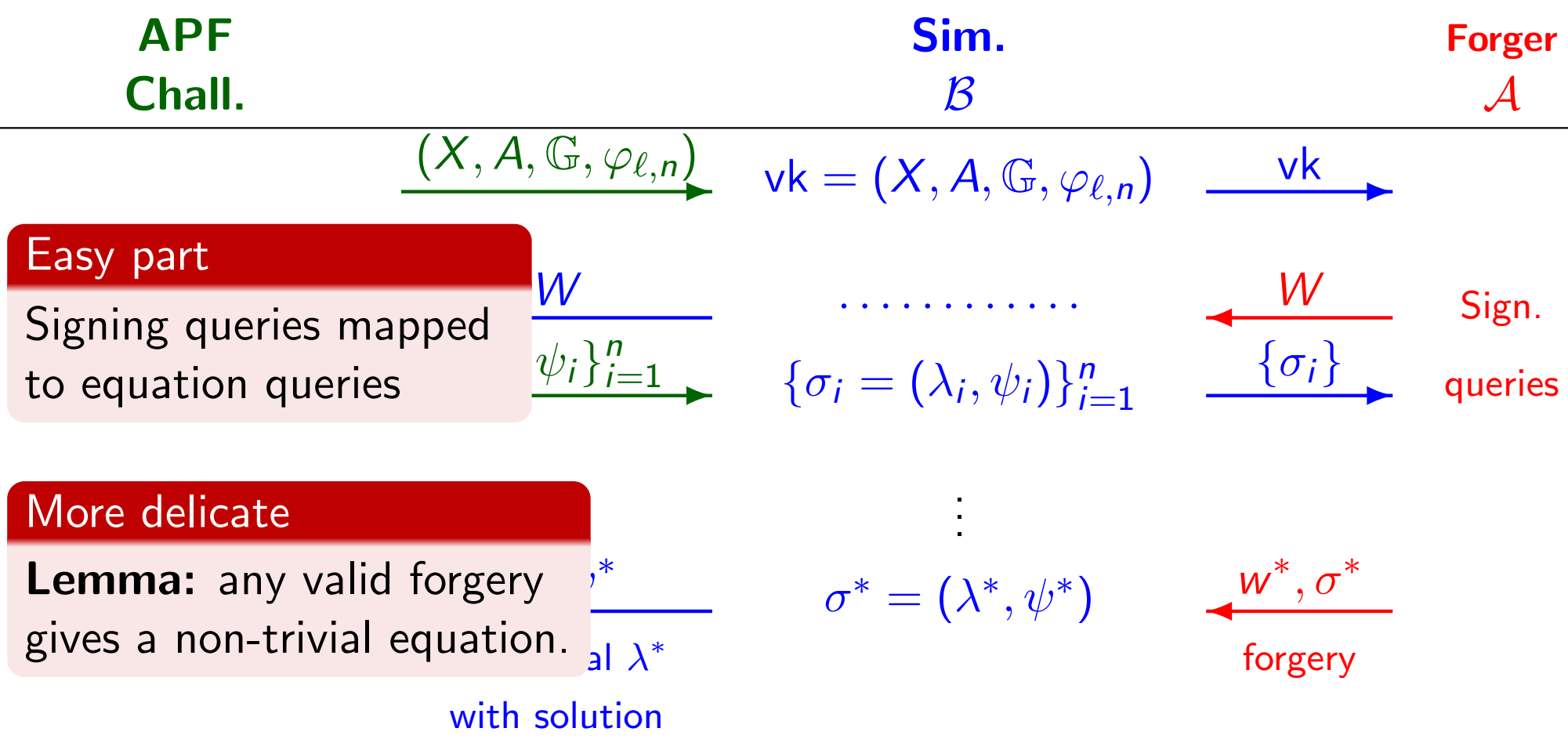
Proof (sketch)



Proof (sketch)



Proof (sketch)



Easy part

Signing queries mapped to equation queries

More delicate

Lemma: any valid forgery gives a non-trivial equation.

Standard signatures from APF groups

NetPFSig(1) $\Rightarrow$ PFSig

- Any Network Coding Signature for $n = 1$ is (syntactically) a signature scheme

Standard signatures from APF groups

NetPFSig(1) $\Rightarrow$ PFSig

- Any Network Coding Signature for $n = 1$ is (syntactically) a signature scheme
- Unfortunately, it allows for some kind of malleability
- Set $Sign(\mathsf{sk}, m) = \mathsf{NetSign}(\mathsf{sk}, 1||m)$. Then ask the verification to say that only vectors $(1||w)$ are valid.

Standard signatures from APF groups

NetPFSig(1) $\Rightarrow$ PFSig

- Any Network Coding Signature for $n = 1$ is (syntactically) a signature scheme
- Unfortunately, it allows for some kind of malleability
- Set $Sign(\mathsf{sk}, m) = \mathsf{NetSign}(\mathsf{sk}, 1||m)$. Then ask the verification to say that only vectors $(1||w)$ are valid.

Distribution for $n = 1$: $\varphi_{\ell,1} : \mathbb{Z}_{2^\ell}^{n'} \rightarrow \mathbb{Z}^{1+m} \times \{0,1\}^*$

- Let fid be a binary string taken with high-entropy distribution;
- $e = H(\mathsf{fid})$ where H is a map to primes
- $s_1 = 1$
- $s_k \in \mathbb{Z}_e$ according to (arbitrary) D_k, $\forall k = 2, \ldots, m$
- Output $(e, \mathsf{fid}, \vec{s})$.

The RSA group is adaptive pseudo-free

We define a concrete parametric distribution $\hat{\varphi} \subset \varphi_{\ell,n}$

$\hat{\varphi} : \mathbb{Z}_{2^\ell}^{n'+n} \longrightarrow \mathbb{Z}^{1+mn} \times \{0,1\}^*$

- Let fid be a random binary string;
- $e = H(\text{fid})$ where H is a map to primes
- For $i = 1$ to n define $\vec{s}^{(i)}$ as follows:
  - $s_i^{(i)} = 1$, $s_j^{(i)} = 0$, $\forall j = 1, \ldots, n : j \neq i$
  - $s_{n+1}^{(i)} \in \mathbb{Z}_e$ with the uniform distribution
  - $s_k^{(i)} \in \mathbb{Z}_e$ according to (arbitrary) D_k, $\forall k = n+2, \ldots, m$
- Output $(e, \text{fid}, \{\vec{s}^{(i)}\}_{i=1}^n)$. $\varphi_{\ell,n}$ defines n equations.

Theorem (RSA is APF)

If the Strong RSA Assumption holds, then $\mathbb{Z}_N^*$ is APF w.r.t. $\hat{\varphi}$.

A framework for Strong RSA-based signatures

- Abstract away almost all known Strong-RSA signatures
- Each scheme can be obtained by appropriately instantiating the parametric distribution $\hat{\varphi}$ (for $n = 1$)
- Note: each scheme does not need a new $\hat{\varphi}$, but only a special case of the one for which our proofs hold.

Which schemes?

- [GHR99]: $x^e = a_1$ $(e = H(M))$
- [CS99]: $x^e = a_1 a_2^{H(C)}$, $C = ChamCommit(M; r)$
- [CL02]: $x^e = a_1 a_2^s a_3^M$
- [Fischlin03]: $x^e = a_1 a_2^s a_3^{s \oplus M}$
- [HK08]: $x^e = a_0 a_1^{M_1} \cdots a_m^{M_m}$ $(M = M_1 \cdots M_m)$ (special case)

A new network coding signature in the standard model

- We obtain a new network coding signature in the standard model based on the Strong-RSA assumption.
- All other known solutions were in the RO model.
- Independently, Attrapadung and Libert [PKC'11] built a standard model scheme in bilinear groups of composite-order.

Conclusions

Summary of our results

- Notion of APF Groups
- Applications of APF: Network Coding/Standard Signatures [a]
- Existence of APF Groups: the RSA group is APF
- A new network coding signature in the standard model
- A better understanding of previous signatures
  Other interpretations:
- Distilling the core of Strong-RSA-based proofs
- (Towards) establishing a connection between provably-secure crypto and formal methods

[a] Current version has a different presentation of network coding/standard signatures from APF

Conclusions

Open problems and follow-up works

- New network coding signatures based on Strong-RSA, qSDH, CDH
- Defining pseudo-freeness for other computational groups
- Supporting more generic parametric distributions
- Other crypto applications of APF groups

Thanks!

Questions?

D. Catalano, D. Fiore, B. Warinschi. *Adaptive Pseudo-Free Groups and Applications*. [Eurocrypt 2011]

Full version at http://eprint.iacr.org/2011/053